OIPE JC185
SEP 1 3 2004
PATENT & TRADEMARK OFFICE

Attorney Docket No. P001-101

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant:	Martin Shiu
Serial No.:	10/003,737
Filed:	November 2, 2001
For:	SYSTEM FOR CONFIGURATION PROGRAMMING
Examiner:	Chameli Das
Art Unit:	2122

Certificate of Mailing Pursuant to 37 C.F.R. §1.8

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to Mail Stop Amendment, Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450 on this 9th day of September, 2004.

[illegible]
Brett N. Dorny, Reg. No. 35,860

September 9, 2004
Boston, Massachusetts

Mail Stop Amendment
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

DECLARATION UNDER 37 CFR §1.131

I, Martin Shiu, hereby declare as follows:

1. I am in the sole inventor and the applicant of the above identified patent application. I make this declaration in support of the application. All facts are made of my own knowledge.

2. The root concept of the invention is original from the software engineer project during my graduate school year at University of Massachusetts Boston in 1988. That project involved into creating a configuration programming language to simplify the software development effort for other project teams. The effort to continue simply the software development processes is continue through 1990 to 1995. I conceived of the key aspects of the invention in 1996. Specifically, I originally conceived of a software development system with improved efficiency and adaptability. I realized that all data could be represented and described by a finite set of basic data model types. By creating code for performing functions which can recognize and operate on these basic data model types, the code can automatically adapt to changes in a data model. A computer application can be created by forming a flow of the functional code.

Serial No.: 10/003,737 **Attorney Docket No.** P001-101

Art Unit: 2122

Attached as Exhibit A are a versioned technical white paper initiated as early as 1994 describes many detail associate with this invention. Some of the high-level picture in that documents has been discussed with my colleagues since that time too. They are available for confirming my statement.

3. On June 10, 1996, I formed a company, Power Object Group, Inc. to create a development system based upon my ideas. For a development system, a large number of functional modules, which I called service objects, were needed. These had to be created in order to show that my ideas would work.

4. In 2000, I moved the company to a new location and added employees. We then began development of a Java based system. In late 2000, we have a functional system and began using it for software development. We continued to augment, improve and modify the system since that date.

I declare under penalty of perjury that the foregoing is true and correct.

Martin Shiu 8/30/04

Martin Shiu

EXHIBIT A

Rapid Application Builder (RAB) vs. CAP2.0

By: Martin Shiu

Rapid Application Builder (RAB)

By: Martin Shiu

What is RAB

- Application development toolkit
- Class library management
- Drag and drop interface
- Provide application design guide line
- Provide application implementation models
- Object based data transaction
- Provide Standard Data Dictionary Interface

What is RAB (Cont.)

- Data mapping manager
- Report manager
- Provide standard client/server models
- Generate/Maintain static application code

What’s **<u>not</u>** the foucs of RAB

- Provide complete solution for an application

Key Benefits

- Automate the development of static but time consumming application process
- Standard application architecture guide line
- Models based development
- Fucused development
- Parallel development
- Accumulate re-usable objects

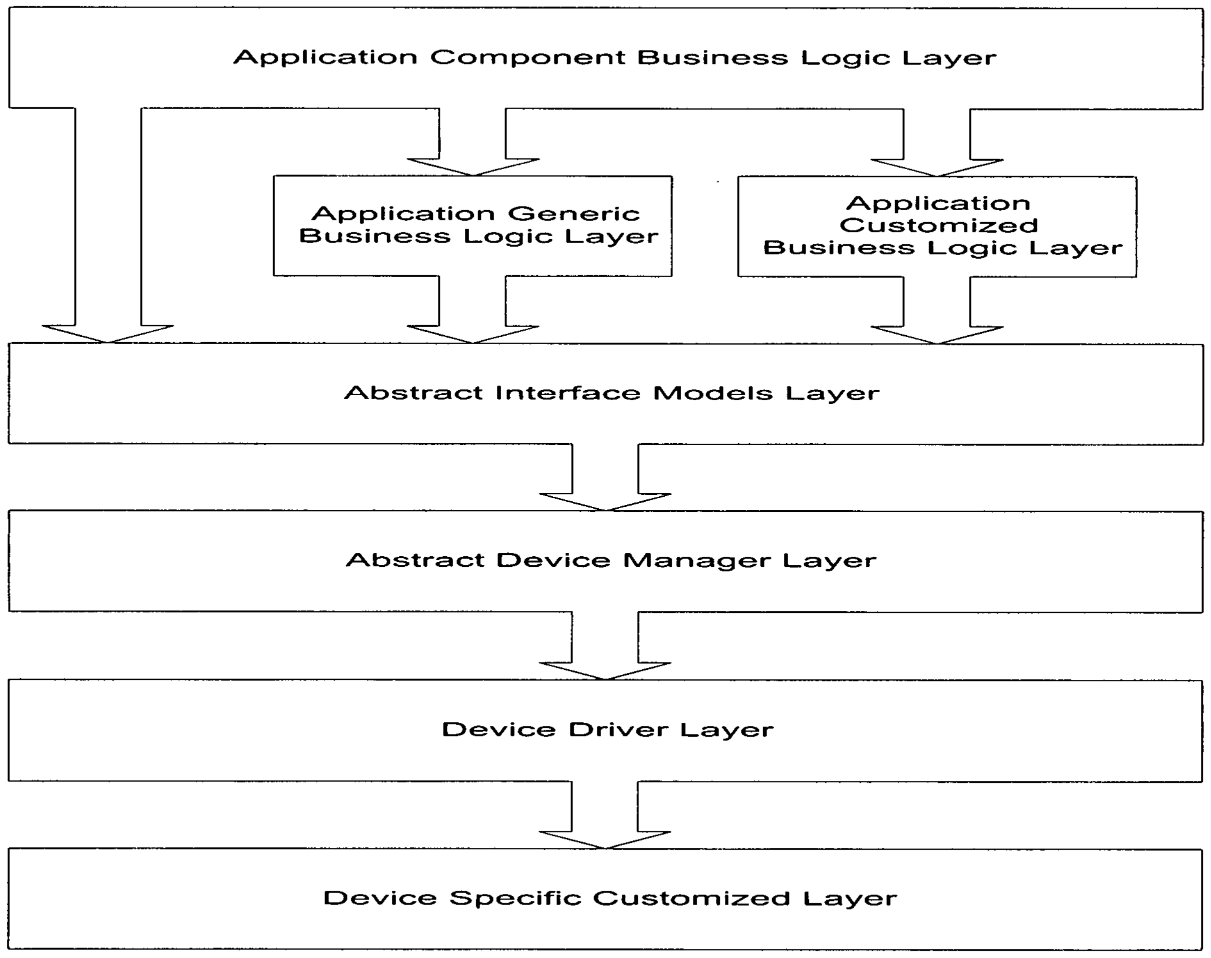
Key Benefits (Cont.)

- Generator/Maintain application code
- Shorten application development time frame
- Simplified maintenance process
- Develop highly portable code
- Maximize usage of third party software

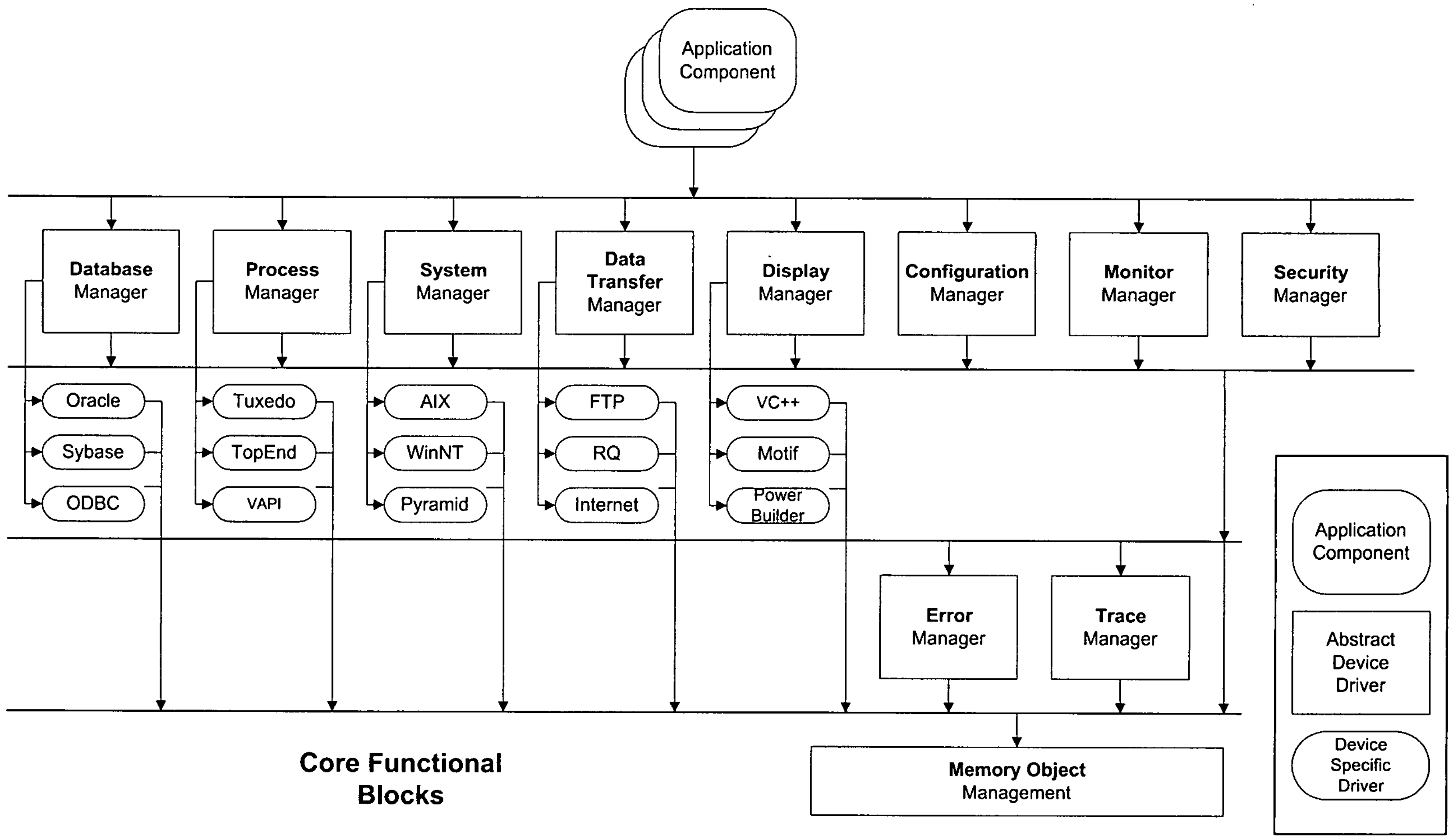
Architecture Guide Line

- Standardize the applications architecture
- Highly Object Oriented driven
- Layered Hierarchy
- Sub-System Management
- Multi-Layers Plug and Play

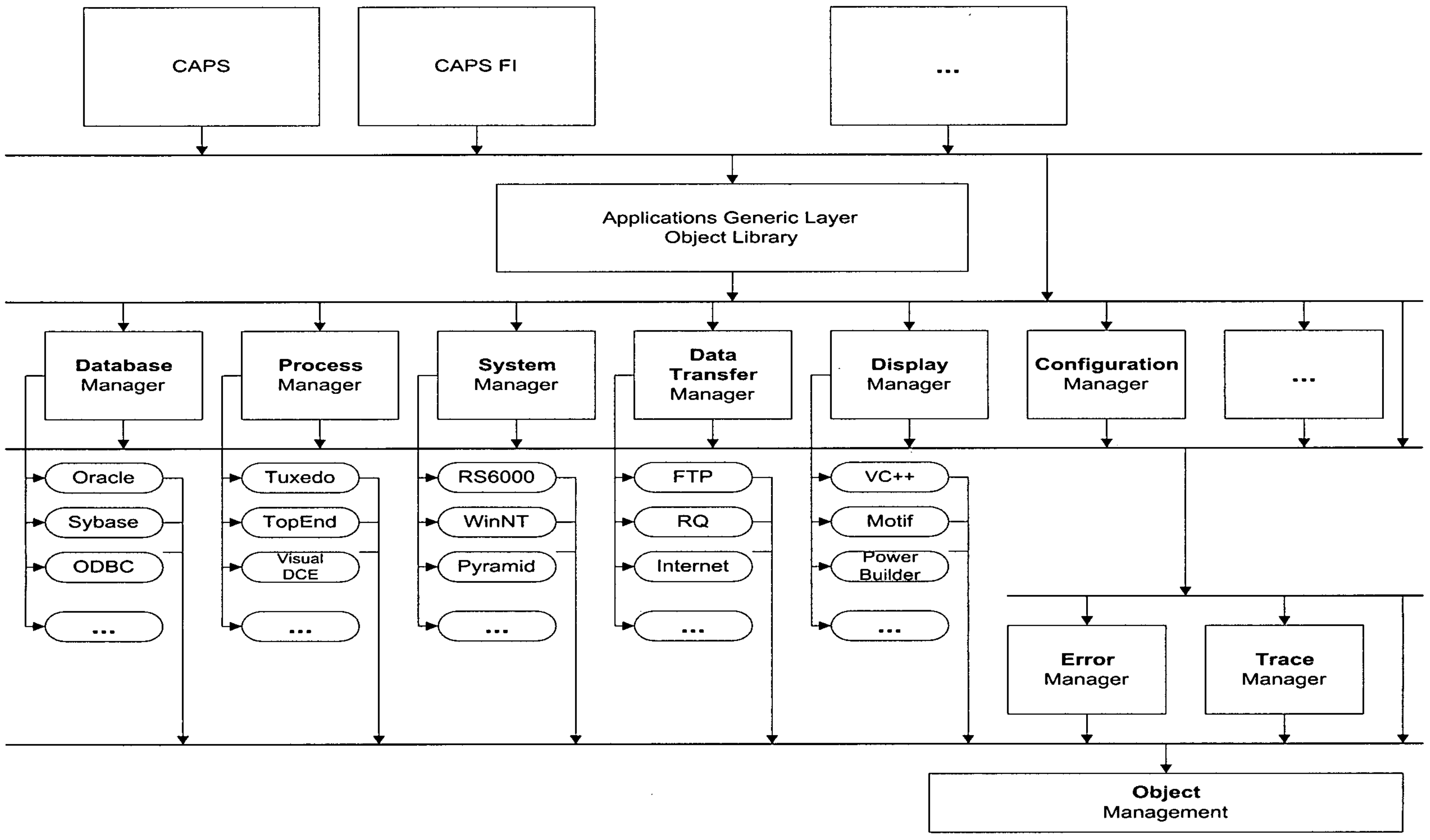
Layer Hierarchy Architecture


Application Component Business Logic Layer
Application Generic Business Logic Layer
Application Customized Business Logic Layer
Abstract Interface Models Layer
Abstract Device Manager Layer
Device Driver Layer
Device Specific Customized Layer

Sub-System/Device Managers



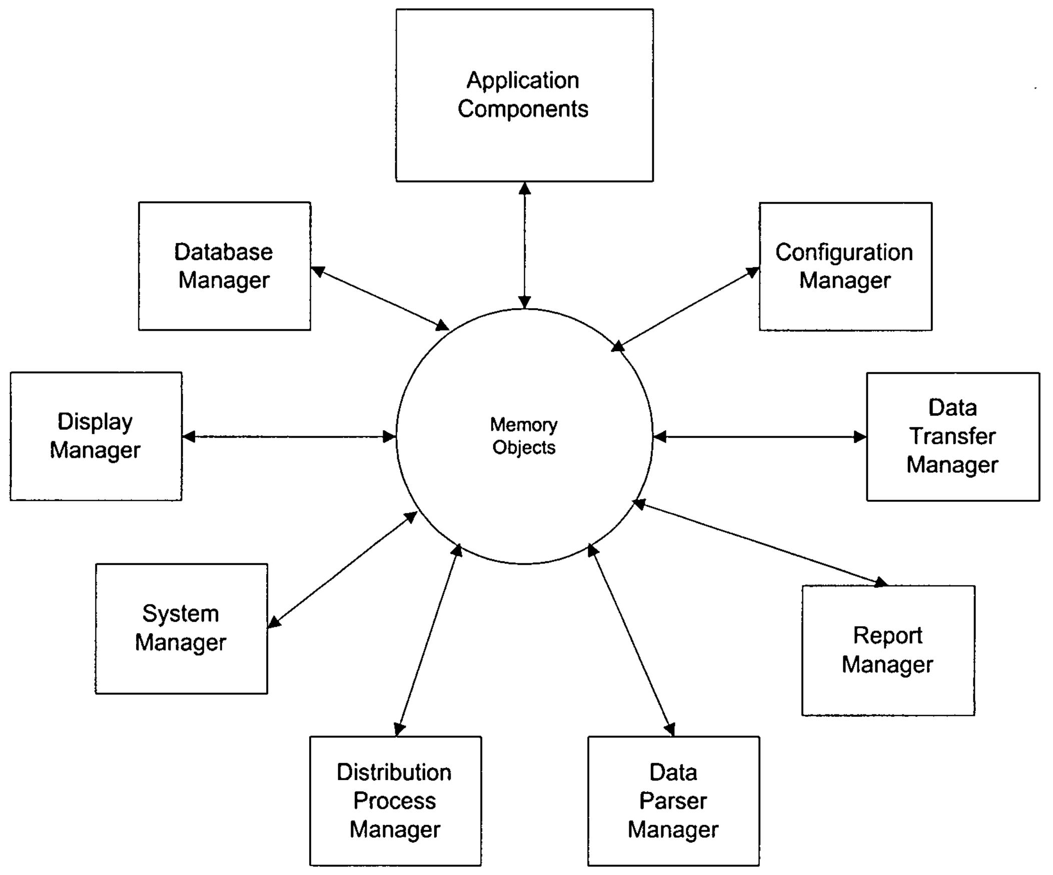
Mutilple Layers Plug and Play



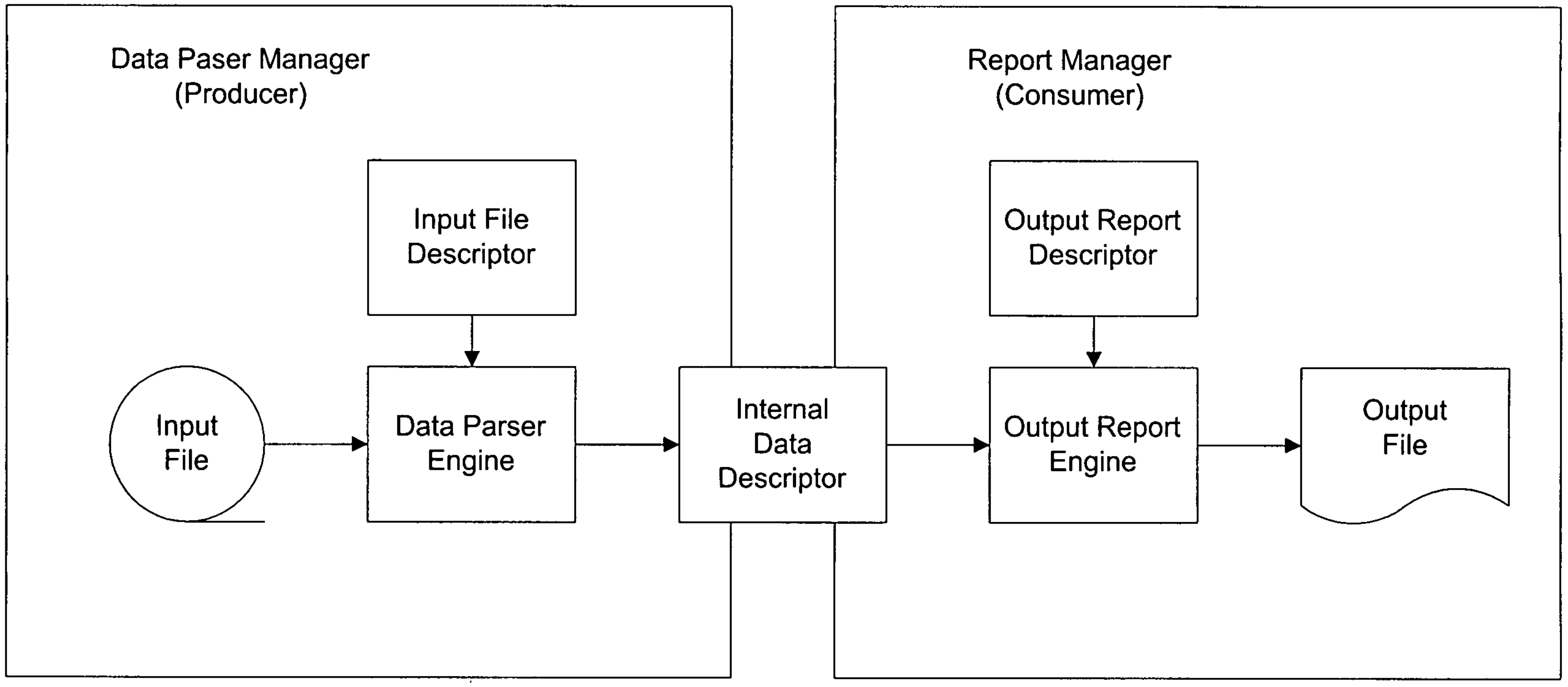
Implementation Guide Line

- Memory based programming
- Vertical programming
- Container based programming

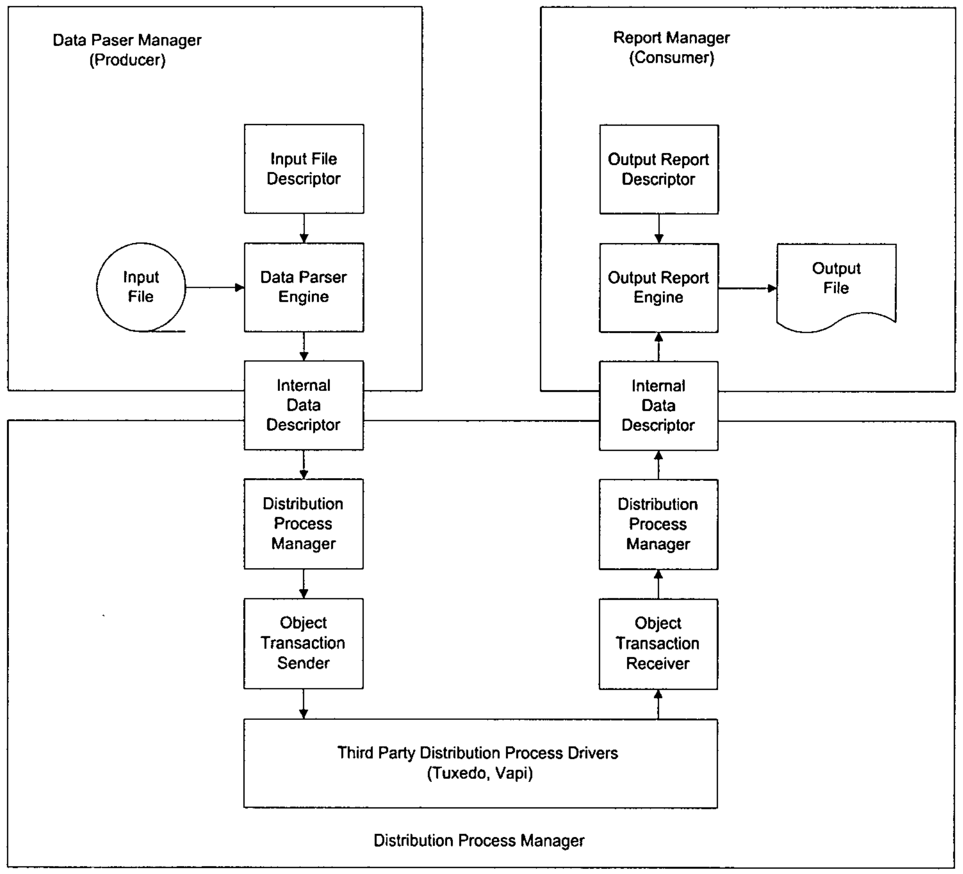
Memory based programming



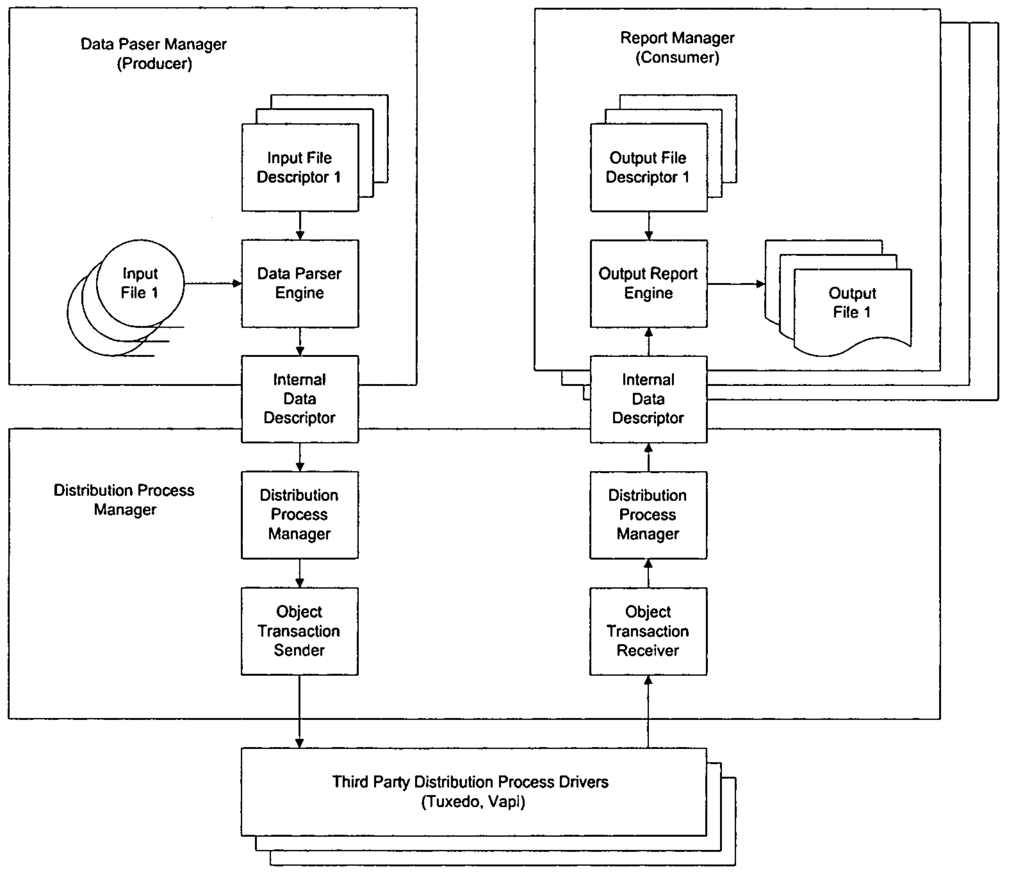
Basic process model



Sync Distrbution Process Model



Async Distribution Process Model


Data Paser Manager (Producer)
Input File Descriptor 1
Input File 1
Data Parser Engine
Internal Data Descriptor
Report Manager (Consumer)
Output File Descriptor 1
Output Report Engine
Output File 1
Internal Data Descriptor
Distribution Process Manager
Distribution Process Manager
Object Transaction Sender
Distribution Process Manager
Object Transaction Receiver
Third Party Distribution Process Drivers (Tuxedo, Vapi)